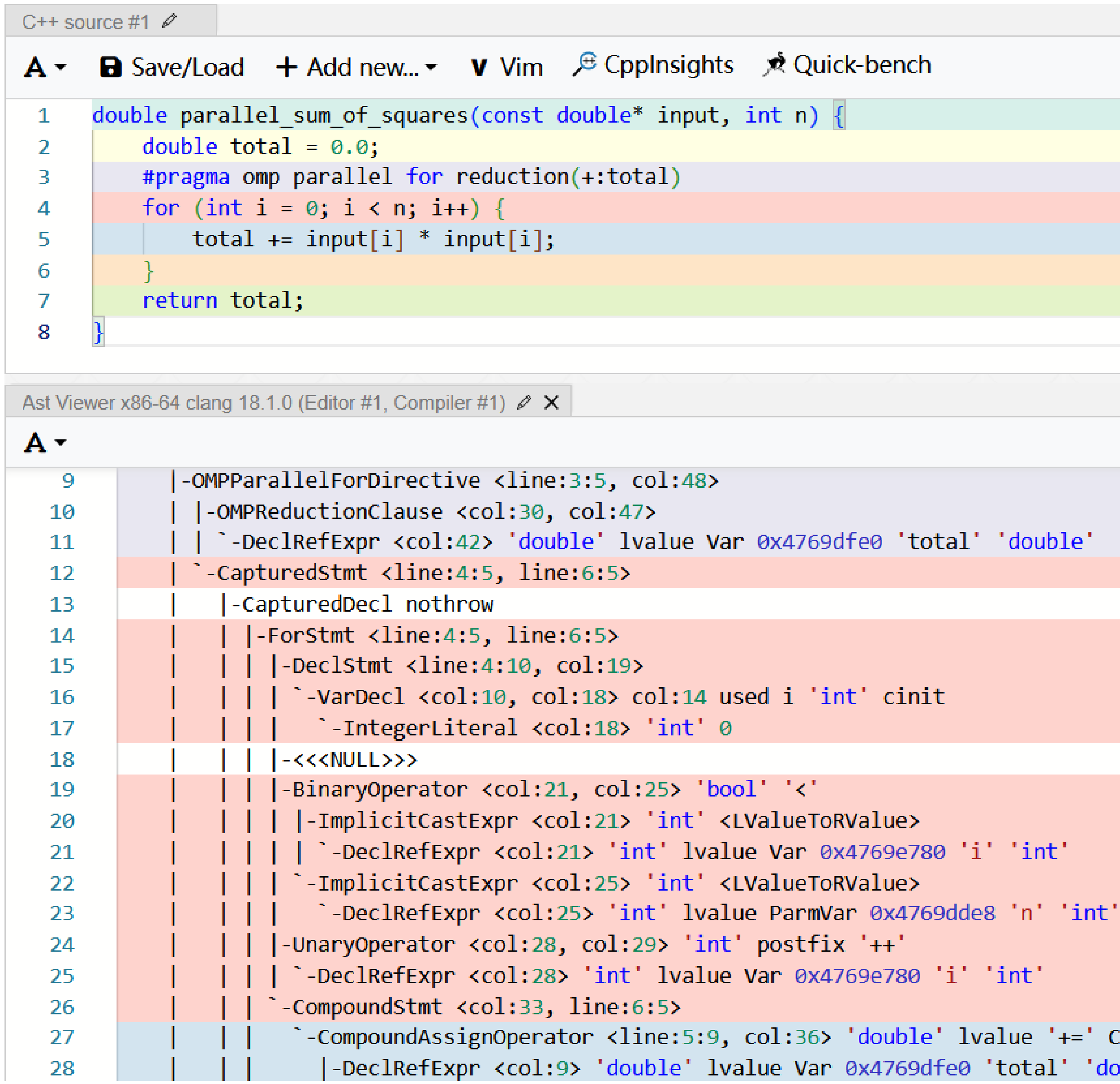
Enable automatic differentiation of OpenMP programs with Clad

Contributor: Jiayang Li

Mentors: Vassil Vassilev, Martin Vassilev

Project Principle

- OpenMP pragmas are converted into Clang AST nodes at compile time
- Therefore, we can add logic to handle OpenMP, just as we traverse other Clang AST nodes to perform differentiation.



```cpp
double parallel_sum_of_squares(const double* input, int n) {
    double total = 0.0;
    #pragma omp parallel for reduction(+:total)
    for (int i = 0; i < n; i++) {
        total += input[i] * input[i];
    }
    return total;
}
```

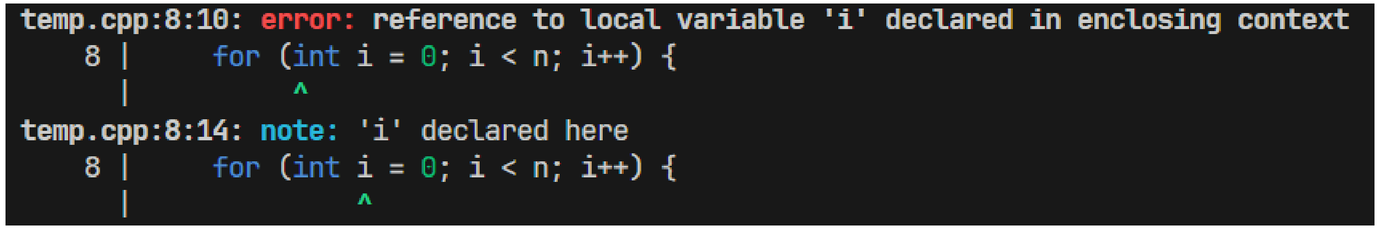
```
|-OMPParallelForDirective <line:3:5, col:48>
| |-OMPReductionClause <col:30, col:47>
| | `-DeclRefExpr <col:42> 'double' lvalue Var 0x4769dfe0 'total' 'double'
| `-CapturedStmt <line:4:5, line:6:5>
|   |-CapturedDecl nothrow
|   | |-ForStmt <line:4:5, line:6:5>
|   | | |-DeclStmt <line:4:10, col:19>
|   | | | `-VarDecl <col:10, col:18> col:14 used i 'int' cinit
|   | | |   `-IntegerLiteral <col:18> 'int' 0
|   | | |-<<<NULL>>>
|   | | |-BinaryOperator <col:21, col:25> 'bool' '<'
|   | | | |-ImplicitCastExpr <col:21> 'int' <LValueToRValue>
|   | | | | `-DeclRefExpr <col:21> 'int' lvalue Var 0x4769e780 'i' 'int'
|   | | | `-ImplicitCastExpr <col:25> 'int' <LValueToRValue>
|   | | |   `-DeclRefExpr <col:25> 'int' lvalue ParmVar 0x4769dde8 'n' 'int'
|   | | |-UnaryOperator <col:28, col:29> 'int' postfix '++'
|   | | | `-DeclRefExpr <col:28> 'int' lvalue Var 0x4769e780 'i' 'int'
|   | | `-CompoundStmt <col:33, line:6:5>
|   | |   `-CompoundAssignOperator <line:5:9, col:36> 'double' lvalue '+=' C
|   | |     |-DeclRefExpr <col:9> 'double' lvalue Var 0x4769dfe0 'total' 'do
```

Done Work

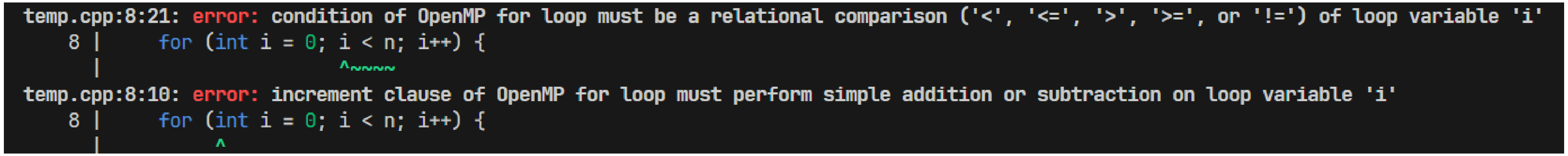
- Correct processing logic for the forward-mode ReductionClause
    - The derivative variable must maintain the same scope as the original variable, so it needs to be added to the same clause.
    - This logic can be reused for other clauses.
- General handling logic for the forward-mode OMPParallelForDirective (still has bugs)
    - Simply recursively transform the loop body.

Current Issues

- The Stmt returned by VisitForStmt is not a standard ForStmt.
  - Current workaround: Directly extract the child ForStmt and pass it to subsequent functions.
- Various semantic issues are likely caused by CapturedDecl.



```
temp.cpp:8:10: error: reference to local variable 'i' declared in enclosing context
    8 |     for (int i = 0; i < n; i++) {
      |          ^
temp.cpp:8:14: note: 'i' declared here
    8 |     for (int i = 0; i < n; i++) {
      |              ^
```



```
temp.cpp:8:21: error: condition of OpenMP for loop must be a relational comparison ('<', '<=', '>', '>=', or '!=') of loop variable 'i'
    8 |     for (int i = 0; i < n; i++) {
      |                     ^~~~~
temp.cpp:8:10: error: increment clause of OpenMP for loop must perform simple addition or subtraction on loop variable 'i'
    8 |     for (int i = 0; i < n; i++) {
      |          ^
```

Next Steps

- Complete the forward-mode code, covering all key OpenMP-related AST nodes.
- Implement and refine the reverse-mode.
- Develop comprehensive tests.